natural

Version 4.1.2 for Mainframes | System Variables

SOFTWARE AG

This document applies to Natural Version 4.1.2 for Mainframes and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Table of Contents

System Variables

This document describes the Natural system variables.

Natural system variables contain information about the current Natural session, such as: the current library, the user and terminal identification; the current status of a loop processing; the current report processing status; the current date and time. They may be referenced at any point within a Natural program.

- System Variables Grouped by Function
  - Application Related System Variables
  - Date and Time System Variables
  - Input/Ouput Related System Variables
  - Natural Environment Related System Variables
  - System Environment Related System Variables
  - XML Related System Variables

See also:

- System Variables in the Natural Programming Guide
- Example of System Variables and System Functions in the Natural Programming Guide

Application Related System Variables

The following Natural system variables are covered:

- *APPLIC-ID
- *APPLIC-NAME
- *COM
- *CONTROL
- *CONVID
- *COUNTER
- *CPU-TIME
- *CURRENT-UNIT
- *DATA
- *DIALOG-ID
- *ERROR-LINE
- *ERROR-NR
- *ERROR-TA
- *ETID
- *EVENT
- *ISN
- *LBOUND
- *LENGTH
- *LEVEL
- *LIBRARY-ID
- *LINE
- *NUMBER
- *OCCURRENCE
- *PROGRAM
- *ROWCOUNT
- *STARTUP
- *STEPLIB
- *SUBROUTINE
- *THIS-OBJECT
- *TYPE
- *UBOUND

Content modifiable:
In the following text, this indicates whether in a Natural program you can assign another value to the system variable, that is, overwrite its content as generated by Natural.

---

*APPLIC-ID

Format/length:	A8
Content modifiable:	No

This system variable contains the ID of the library to which the user is currently logged on.

*APPLIC-NAME

Format/length:	A32
Content modifiable:	No

If Natural Security is installed, this system variable contains the name of the library to which the user is logged on.

If Natural Security is not installed, this system variable contains the name "SYSTEM".

*COM

Format/length:	A128
Content modifiable:	Yes

This system variable designates a communication area which can be used to process data from outside a screen window.

Normally when a window is active, no data can be entered on the screen outside the window. However, if a map contains *COM as a modifiable field, that field will still be available for the user to enter data when a window is currently on the screen. Further processing can then be made dependent on the content of *COM. This allows you to implement user interfaces where a user can always enter data in the command line, even when a window with its own input fields is active.

Note:
Although *COM can be used as a modifiable field in an INPUT statement, it is **not** treated as an input field, but as a system variable; that is, any input entered into the *COM field will be taken as it is, without any input processing (e.g. conversion to upper case) being performed on it.
Once *COM has been displayed on the screen via an INPUT statement, every subsequent INPUT or REINPUT statement will cause the current content of *COM to be displayed.

See also Dialog Design, Processing Data Outside an Active Window, in the Natural Programming Guide.

*CONTROL

This system variable is only available under Windows 98 and Windows NT/2000.

Format/length:	Handle
Content modifiable:	No

This system variable contains the handle of the dialog element for which the current event has been triggered.

For details on events, see Event-Driven Programming Techniques in the Natural Programming Guide.

*CONVID

Format/length:	I4
Content modifiable:	Yes

This system variable contains the conversation ID of the current conversational remote procedure call (RPC). This ID is set by an OPEN CONVERSATION statement.

Via an OPEN CONVERSATION statement, a client can get a server for exclusive use to execute a number of services (subprograms) within one server process. This exclusive use is called conversation. The OPEN CONVERSATION statement is used to open a conversation and specify the subprograms to be involved in this conversation. When an OPEN CONVERSATION statement is executed, it assigns a unique ID which identifies the conversation to the system variable *CONVID.

Several conversations can be open at the same time. To switch from one open conversation to another, you assign the corresponding conversation ID to *CONVID.

For further information on Natural RPC, see the Natural RPC documentation.

*COUNTER

Format/length:	P10
Content modifiable:	Yes

This system variable contains the number of times a processing loop initiated by a FIND, READ or HISTOGRAM statement has been entered.

(r) in parentheses after *COUNTER is used to indicate the statement label or source-code line number of the FIND, READ, or HISTOGRAM. If (r) is not specified, *COUNTER represents the number of times the currently active processing loop has been entered.

*COUNTER is not incremented if a record is rejected as a result of the criteria specified in a WHERE clause. *COUNTER is incremented if a record is rejected as a result of an ACCEPT/REJECT statement.

*CPU-TIME

Format/length:	I4
Content modifiable:	No

*CPU-TIME contains the CPU time currently used by the Natural process in units of 10 ms.

This system variable always contains the value zero for the following operating or TP monitor systems on mainframe computers:

- VSE/ESA
- CICS
- IMS/TM
- UTM
- Com-plete (versions lower than 6.3)

*CURRENT-UNIT

This system variable is only available under Windows and UNIX.

Format/length:	A32
Content modifiable:	No

This system variable contains the name of the currently executed unit. This is

- the function name in case of the object type FUNCTION,
- the subroutine name in case of the object type SUBROUTINE, see also *SUBROUTINE,
- the object name in case of all other object types (program, subprogram, map, dialog, etc.); see also *PROGRAM.

The contents of *CURRENT-UNIT will always be in upper case.

*DATA

Format/length:	N3
Content modifiable:	No

This system variable contains the number of data elements in the Natural stack which are available to the next INPUT statement as input data. *DATA will contain "0" when the stack is empty. A value of "-1" indicates the next element in the stack is a command or the name of a Natural transaction.

The settings of the IA and ID parameters at the time of execution of the STACK statement are used to determine the *DATA value.

*DIALOG-ID

This system variable is only available under Windows.

Format/length:	I4
Content modifiable:	No

This system variable contains the ID of the current instance of a dialog.

For details on dialogs, see Event-Driven Programming Techniques in the Natural Programming Guide.

*ERROR-LINE

Format/length:	N4
Content modifiable:	No

This system variable contains the source-code line number of the statement that caused an error.

*ERROR-NR

Format/length:	N7
Content modifiable:	Yes

This system variable contains the error number of the error which caused an ON ERROR condition to be entered.

Only error numbers in the range from 0 to 9999 are supported.

Normally, *ERROR-NR contains the Natural *system* error number which caused an error condition to be entered; however, when a "REINPUT WITH TEXT *nnnn" statement is executed, the *application-specific* message number *nnnn* is placed into *ERROR-NR.

You may modify the content of this system variable via a Natural program; however, not within an ON ERROR statement block.

*ERROR-NR is reset to "0" when another level 1 program is executed.

*ERROR-TA

Format/length:	A8
Content modifiable:	Yes

This system variable contains the name of the program which is to receive control in the event of an error condition.

When an error occurs, Natural will execute a STACK TOP DATA statement and place at the top of the stack the following information, which can be used as INPUT data by an error transaction: Error number (N4 if SG=OFF; N5 if SG=ON), Line number (N4), Status (A1), Program name (A8), Level (N2). If the Status is either "C" or "L", the Line number will be "0". The Status may be one of the following:

C	Command processing error.
L	Logon error.
O	Object time error.
S	Non-correctable Syntax error.
R	Error on Remote server (in conjunction with Natural RPC).

*ETID

Format/length:	A8
Content modifiable:	No

This system variable contains the current identifier of transaction data for Adabas. The default value is one of the following:

- the value of the Natural profile parameter ETID,
- the user ID as passed from the TP monitor (on mainframe computers only),
- the value provided in the user exit during Natural initialization (on mainframe computers only),
- the value from the security profile of the currently active user (applies only under Natural Security).

*EVENT

This system variable is only available under Windows.

Format/length:	A32
Content modifiable:	No

This system variable contains the name of the current event.

For details on events, see Event-Driven Programming Techniques in the Natural Programming Guide.

*ISN

Format/length:	P10
Content modifiable:	Yes

This system variable contains the Adabas internal sequence number (ISN) of the record currently being processed within a processing loop initiated by a FIND or READ statement.

(r) in parentheses after *ISN is used to indicate the label or statement number of the statement in which the FIND or READ was issued. If (r) is not specified, *ISN represents the ISN of the record currently being processed in the currently active processing loop.

For the HISTOGRAM statement, *ISN contains the number of the occurrence in which the descriptor value last read is contained (*ISN = 0 if the descriptor is not contained within a periodic group).

*ISN for Tamino

*ISN contains the XML object ID.

*ISN for DL/I and SQL Databases

*ISN cannot be used for DL/I and SQL databases.

*ISN for VSAM

For VSAM databases, *ISN can only be applied to ESDS and RRDS. For ESDS, *ISN contains the relative byte address (RBA) and for RRDS the relative record number (RRN) of the record currently being processed within a processing loop initiated by a FIND or READ statement.

*LBOUND

This system variable is only available under Windows and UNIX.

Format/length:	I4
Content modifiable:	No

LBOUND contains the current lower boundary (index value) of an array for the specified dimension(s) (1, 2 or 3) or for all dimensions (notation).

Syntax:

```
*LBOUND (operand1 [,dim])
```

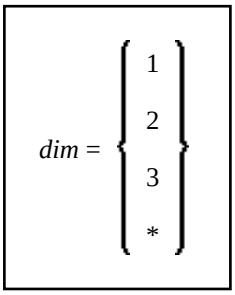
operand1

operand1 is the array for which the lower boundary is specified. The index notation of the array is optional. As index notation only the complete range notation * is allowed for each dimension.

Operand	Possible Structure					Possible Formats												Referencing Permitted	Dynamic Definition
Operand1			A			A	N	P	I	F	B	D	T	L	C	G	O	yes	no

dim

dim is the dimension number for which the current lower boundary is returned:



Explanation:

If no dimension is specified, the lower bound of the first dimension is returned.

If 1, 2 or 3 is specified, the lower bound of the first, second or third dimension is returned.

If * is specified, the lower bound of all the defined dimensions are returned, that is

- 1 in case of an one dimensional array,
- 2 in case of a two dimensional array,
- 3 in case of three dimensional array.

If an X-array is not allocated, the lower bound of this X-array is undefined and *LBOUND leads to a runtime error. In order to avoid the runtime error, *OCCURRENCE may be used to check against zero occurrences:

```
DEFINE DATA LOCAL
  1 #XA(A5/1:*)
END-DEFINE
IF *OCCURRENCE (#XA) NE 0 AND *LBOUND(#XA) > 10
  THEN ...
```

Examples:

```
DEFINE DATA LOCAL
  1 #I  (I4)
  1 #J  (I4/1:3)
  1 #XA (A5/10:*,20:*)
END-DEFINE
#I  := *LBOUND(#XA)       /* lower bound of 1st dimension is 10
#I  := *LBOUND(#XA,1)     /* lower bound of 1st dimension is 20
#I  := *LBOUND(#XA,2)     /* lower bound of 2nd dimension is 20
#J(1:2):= *LBOUND(#XA,*)  /* lower bound of all dimensions
                          /* (1st and 2nd)
                          /* #J(1) is 10 and #J(2) is 20
```


See also *UBOUND and *OCCURRENCE.

LENGTH(field)*

Format/length:	I4
Content modifiable:	No

This system variable returns the currently used length (in bytes) of a field defined as a dynamic variable. *LENGTH*(field)* applies to dynamic variables only.

*LEVEL

Format/length:	N2
Content modifiable:	No

This system variable contains the level number of the program, subprogram, external subroutine, map, helproutine or dialog which is currently active. Level 1 is a main program.

*LEVEL does not apply to inline subroutines.

*LIBRARY-ID

Format/length:	A8
Content modifiable:	No

This system variable contains the current library ID (as specified by the user in the LOGON command).

This variable is the equivalent of the variable *APPLIC-ID.

*LINE

Format/length:	I4
Content modifiable:	No

It contains the number of the line currently executed in a Natural object.

Note:
If a Natural programming object has been compiled with the Natural Optimizer Compiler, the value of *LINE may not reflect the correct line number if the source line where the system variable is used is part of a sequence of statements that have been optimized by the Natural Optimizer Compiler.

*NUMBER

Format/length:	P10
Content modifiable:	Yes

This system variable contains either the number of records which were selected as a result of a FIND statement (as a result of the WITH clause) or the number of values selected as a result of a HISTOGRAM statement.

(r) in parentheses after *NUMBER is used to indicate the statement label or source-code line number of the statement in which the FIND or HISTOGRAM statement was issued. If *(r)* is not specified, *NUMBER represents the number of records selected from the FIND or HISTOGRAM used to initiate the currently active processing loop.

Note:
When the Adabas file accessed is protected by the Adabas facility Security By Value, *NUMBER will contain 9999999999, if more than 1 record was found.
If 1 record was found *NUMBER will contain 1. If no record was found *NUMBER will contain 0.

*NUMBER for DL/I

For DL/I databases, *NUMBER does not contain the number of segment occurrences found. It contains "0" if no segment occurrence satisfies the search criterion, and it contains a value of 8,388,607=X'7FFFFF' if at least one segment occurrence satisfies the search criterion.

*NUMBER for SQL Databases

When used with a FIND NUMBER without a WHERE clause or with a HISTOGRAM statement, *NUMBER will contain the number of rows found. Otherwise, when applied to an SQL database table, *NUMBER will not contain the number of rows found: *NUMBER will be "0" if no rows have been found; any value other than "0" indicates that rows have been found, but the value will have no relation to the number of rows actually found.

If a FIND NUMBER with a WHERE clause is used, the number of rows found is returned in *COUNTER.

*NUMBER for VSAM

For VSAM databases, *NUMBER only contains the number of records found, when used with a HISTOGRAM statement, or with a FIND statement which uses the operator "EQUAL TO" in the search criterion. With any other operator, *NUMBER will not contain the number of records found: *NUMBER will be "0" if no records have been found; any other value indicates that records have been found, but the value will have no relation to the number of records actually found.

*NUMBER for XML Databases

When used with a FIND NUMBER statement without a WHERE clause, *NUMBER will contain the number of rows found. Otherwise, when applied to an XML database, *NUMBER will not contain the number of rows found: *NUMBER will be "0" if no rows have been found. Any value other than "0" indicates that rows have been found. However, the value will have no relation to the number of rows actually found.

If a FIND NUMBER with a WHERE clause is used, the number of rows found is returned in *COUNTER

*OCCURRENCE

Format/length:	I4
Content modifiable:	No

This system variable contains the current number of occurrences of an array for the specified dimension(s).

Syntax:

```
*OCCURRENCE (operand1 [,dim])
```

Note:
*OCCURRENCE may be abbreviated as *OCC.

operand1

operand1 is the array for which the number of occurrences is returned. The index notation of the array is optional. As index notation only the complete range notation * is allowed for each dimension.

Operand	Possible Structure					Possible Formats											Referencing Permitted	Dynamic Definition
Operand1			A			A	N	P	I	F	B	D	T	L	C	G	O	yes	no

dim

dim is the dimension number for which the current number of occurrences is returned:

dim = { 1 | 2 | 3 | * }

Explanation:

1 One-dimensional array. This is the default, if *dim* is not specified.

2 Two-dimensional array.

3 Three-dimensional array.

* All dimensions defined for the corresponding array apply.

In a parameter data area, you can use the index notation "1:V" to define an array with a variable number of occurrences (see the DEFINE DATA statement in the Natural Statements documentation). The current number of occurrences of such an array is determined at runtime. With *OCCURRENCE, you can ascertain the current number of array occurrences.

Example:


```
DEFINE DATA
  PARAMETER
    1 #ARRAY (A5/1:V)
  LOCAL
    1 #I (I4)
 ...
END-DEFINE
...
FOR #I = 1 TO *OCCURRENCE(#ARRAY)
  ...
END-FOR
...
```

For further examples, see programs/subprograms OCC1P and OCC1N, as well as OCC2P and OCC2N in library SYSEXRM.

*PROGRAM

Format/length:	A8
Content modifiable:	No

This system variable contains the name of the Natural object that is currently being executed.

*ROWCOUNT

Format/length:	I4
Content modifiable:	No

This system variable contains the number of rows that were deleted, updated or inserted by one of the Natural SQL statements "searched" DELETE, "searched" UPDATE or INSERT (with *select-expression*) respectively. *ROWCOUNT always refers to the last executed one of these statements.

*STARTUP

Format/length:	A8
Content modifiable:	Yes

The program whose name is contained in this system variable will be executed whenever Natural would otherwise display the command input prompt (NEXT prompt or Direct Command line/window).

*STARTUP contains the name of the program which has been entered in Natural Security as startup transaction in the security profile of the respective library (except in batch mode; see also the Natural Security documentation).

If no startup transaction is specified, or if Natural Security is not used, *STARTUP contains the value of the profile parameter STARTUP (except on mainframe computers).

On mainframe computers, if no startup transaction is specified or if Natural Security is not used, the value of *STARTUP depends on the setting of the profile parameter MENU:

- If MENU=OFF is set, *STARTUP will be blank.
- If MENU=ON is set, *STARTUP contains "MAINMENU"; that is, the Natural main menu will be displayed.

Via a Natural program, you can assign to *STARTUP a program name which will always overwrite its previous content.

Note:
A startup program used in batch mode must contain a FETCH or STACK COMMAND statement; otherwise error NAT9969 may occur.

If you invoke the command input prompt by entering "%%" (or any equivalent command) - either in a non-SECURITY environment or in a SECURITY environment in which command mode is not prohibited for the current library - the startup mechanism will be deactivated. To subsequently re-activate it, log on to the library again or execute a program which re-assigns the name of a program to *STARTUP.

In a Natural Security environment in which command mode is prohibited for the current library, "%%" will cause the program whose name is contained in *STARTUP to be invoked.

*STEPLIB

Format/length:	A8
Content modifiable:	No

This system variable contains the name of the Steplib library which has been concatenated to the Natural library to which the user is currently logged on.

If Natural Security is not active, *STEPLIB contains:

- under UNIX and Windows: the *STEPLIB name specified with the profile parameter LSTEP in the parameter file used;
- on mainframe computers: the name specified with the profile parameter STEPLIB.

If Natural Security is active, the value may be defined in the security profile of a given library.

Note:
The database ID and file number of the *STEPLIB library are derived from its name. Apart from the library SYSTEM, libraries with the name SYSxxx are assumed to be in FNAT and other libraries are assumed to be in FUSER.

*SUBROUTINE

Format/length:	A32
Content modifiable:	No

This system variable contains the name of the external subroutine that is currently being executed. The contents of *SUBROUTINE will always be in upper case.

*THIS-OBJECT

Format/length: HANDLE OF OBJECT

Content modifiable: No

This system variable contains a handle to the currently active object. The currently active object uses *THIS-OBJECT to either execute its own methods or pass a reference to itself to another object.

*THIS-OBJECT only contains an actual value when a method is being executed. Otherwise it contains NULL-HANDLE.

*TYPE

This system variable is only available under Windows and UNIX.

Format/length: A32

Content modifiable: No

This system variable contains the type of the Natural object which is currently executed.

Valid values of *TYPE:

Value:	Object Type:
FUNCTION	Function
HELPROUTINE	Helproutine
DIALOG	Dialog
MAP	Map
SUBPROGRAM	Subprogram
PROGRAM	Program
SUBROUTINE	Subroutine

*UBOUND

This system variable is only available under Windows and UNIX.

Format/length: I4

Content modifiable: No

UBOUND contains the current upper boundary (index value) of an array for the specified dimension(s) (1, 2 or 3) or for all dimensions (notation).

Syntax:


```
*UBOUND (operand1 [,dim])
```

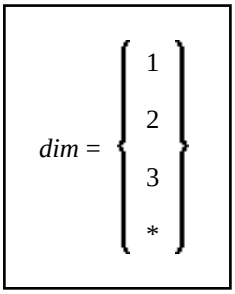
operand1

operand1 is the array for which the upper boundary is specified. The index notation of the array is optional. As index notation only the complete range notation * is allowed for each dimension.

Operand	Possible Structure					Possible Formats												Referencing Permitted	Dynamic Definition
Operand1			A			A	N	P	I	F	B	D	T	L	C	G	O	yes	no

dim

dim is the dimension number for which the current upper boundary is returned:



Explanation:

If no dimension is specified, the upper bound of the first dimension is returned.

If 1, 2 or 3 is specified, the upper bound of the first, second or third dimension is returned.

If * is specified, the upper bound of all the defined dimensions are returned, that is

- 1 in case of an one dimensional array,
- 2 in case of a two dimensional array,
- 3 in case of three dimensional array.

If an X-array is not allocated, the upper bound of this X-array is undefined and *UBOUND leads to a runtime error. In order to avoid the runtime error, *OCCURRENCE may be used to check against zero occurrences:

```
DEFINE DATA LOCAL
  1 #XA(A5/1:*)
END-DEFINE
IF *OCCURRENCE (#XA) NE 0 AND *UBOUND(#XA) > 10
  THEN ...
```

Examples:


```
DEFINE DATA LOCAL
  1 #I  (I4)
  1 #J  (I4/1:3)
  1 #XA (A5/*:10,*:20)
END-DEFINE
#i  := *UBOUND(#XA)        /* upper bound of 1st dimension is 10
#i  := *UBOUND(#XA,1)      /* upper bound of 1st dimension is 10
#i  := *UBOUND(#XA,2)      /* upper bound of 2nd dimension is 20
#j(1:2):= *UBOUND(#XA,*)   /* upper bound of all dimensions
                           /* (1st and 2nd)
                           /* #J(1) is 10 and #J(2) is 20
```

See also *LBOUND and *OCCURRENCE.

Date and Time System Variables

The following topics are covered:

Usage

The date and time system variables listed below may be specified in the following places:

- statements:
  COMPUTE
  DISPLAY
  MOVE
  PRINT
  WRITE
- logical condition criteria

The contents of date and time system variables as generated by Natural are **non-modifiable**, which means that in a Natural program you cannot assign another value to any of them.

Date System Variables

All date system variables contain the current date. The format of the date is different for each date variable, as indicated below.

Date Variable	Format/Length	Date Format*
***DATD**	A8	DD.MM.YY
***DAT4D**	A10	DD.MM.YYYY
***DATE**	A8	DD/MM/YY
***DAT4E**	A10	DD/MM/YYYY
***DATG**	A15	DD*monthname*YYYY (Gregorian date)
***DATI**	A8	YY-MM-DD
***DAT4I**	A10	YYYY-MM-DD
***DATJ**	A5	YYDDD (Julian date)
***DAT4J**	A7	YYYYDDD (Julian date)
***DATN**	N8	YYYYMMDD
***DATU**	A8	MM/DD/YY
***DAT4U**	A10	MM/DD/YYYY
***DATV**	A11	DD-MON-YYYY
***DATVS**	A9	DDMONYYYY
***DATX**	D	internal date format

* D = day, M = month, Y = year, MON = leading three bytes of the month's name as in *DATG

Time System Variables

Time Variable	Format/Length	Explanation
TIMD(r)*	N7	Can only be used in conjunction with a previous SETTIME statement. Contains the time that has elapsed after the SETTIME statement was executed (in format HHIISST (*)). (r) represents the statement label or source-code line number of the SETTIME statement used as the basis for *TIMD.
***TIME**	A10	Contains the time of day in format HH:II:SS.T (*).
***TIME-OUT**	N5	Contains the number of seconds remaining before the current transaction will be timed out (only available with Natural Security). *TIME-OUT is "0" if transaction mode has not been entered. Transaction mode is entered with the execution of a FIND, READ or GET statement that reads a database record for the purpose of updating or deleting the record. *TIME-OUT is reset to "0" when an END TRANSACTION or BACKOUT TRANSACTION statement is executed.
***TIMESTMP**	B8	Machine-internal store clock value.
***TIMN**	N7	Contains the time of day in format HHIISST (*).
***TIMX**	T	Contains the time of day in internal time format.

* H = hour, I = minute, S = second, T = tenth of a second.

Example of Date and Time System Variables

Code Example:

```
* EXAMPLE 'DATIVAR': DATE AND TIME SYSTEM VARIABLES
************************************************
WRITE NOTITLE
  'DATE IN FORMAT DD.MM.YYYY  '   *DAT4D /
  'DATE IN FORMAT DD/MM/YYYY  '   *DAT4E /
  'DATE IN FORMAT DD-MON-YYYY '   *DATV  /
  'DATE IN FORMAT DDMONYYYY   '   *DATVS /
  'DATE IN GREGORIAN FORM     '   *DATG  /
  'DATE IN FORMAT YYYY-MM-DD  '   *DAT4I /
  'DATE IN FORMAT YYYYDDD     '   *DAT4J /
  'DATE IN FORMAT YYYYMMDD    '   *DATN (AD=L) /
  'DATE IN FORMAT MM/DD/YYYY  '   *DAT4U /
  'DATE IN INTERNAL FORMAT    '   *DATX (DF=L) ///
  'TIME IN FORMAT HH:MM:SS.T  '   *TIME /
  'TIME IN FORMAT HHMMSST     '   *TIMN (AD=L) /
  'TIME IN INTERNAL FORMAT    '   *TIMX /
*
MOVE *DATX TO #DATE(D)
ADD 14 TO #DATE
WRITE 'CURRENT DATE'              *DATX (DF=L) 3X
      'CURRENT DATE + 14 DAYS   ' #DATE (DF=L)
MOVE *TIMX TO #TIME(T)
ADD 100 TO #TIME
WRITE 'CURRENT TIME'              *TIMX 5X
      'CURRENT TIME + 10 SECONDS' #TIME
END
```

Screen Display:

```
DATE IN FORMAT DD.MM.YYYY   14.01.2003
DATE IN FORMAT DD/MM/YYYY   14/01/2003
DATE IN FORMAT DD-MON-YYYY  14-Jan-2003
DATE IN FORMAT DDMONYYYY    14Jan2003
DATE IN GREGORIAN FORM      14January  2003
DATE IN FORMAT YYYY-MM-DD   2003-01-14
DATE IN FORMAT YYYYDDD      2003014
DATE IN FORMAT YYYYMMDD     20030114
DATE IN FORMAT MM/DD/YYYY   01/14/2003
DATE IN INTERNAL FORMAT     2003-01-14


TIME IN FORMAT HH:MM:SS.T   10:52:19.0
TIME IN FORMAT HHMMSST      1052190
TIME IN INTERNAL FORMAT     10:52:19

CURRENT DATE 2003-01-14   CURRENT DATE + 14 DAYS    2003-01-28
CURRENT TIME 10:52:19     CURRENT TIME + 10 SECONDS 10:52:29
```


Input/Output Related System Variables

The following system variables are covered:

- *CURS-COL
- *CURS-FIELD
- *CURS-LINE
- *CURSOR
- *LINE-COUNT
- *LINESIZE
- *LOG-LS
- *LOG-PS
- *PAGE-NUMBER
- *PAGESIZE
- *PF-KEY
- *PF-NAME
- *WINDOW-LS
- *WINDOW-POS
- *WINDOW-PS

Content modifiable:
In the following text, this indicates whether in a Natural program you can assign another value to the system variable, that is, overwrite its content as generated by Natural.

---

*CURS-COL

Format/length:	P3
Content modifiable:	Yes (however, a negative value must not be assigned)

This system variable contains the number of the column in which the cursor is currently positioned.

The cursor position is defined within the currently active window, regardless of its physical placement on the screen, starting with position 1/1 from the upper left corner of a logical page.

If the value of *CURS-COL is negative, this indicates that the cursor is outside the active window. If *CURS-COL is negative, *CURS-LINE will also contain a negative value. In this case, the absolute values of both system variables indicate the position of the cursor on the physical screen.

Note:
The message line, function-key lines and infoline/statistics line are not counted as data lines on the screen.

See also Dialog Design, Column-Sensitive Processing, in the Natural Programming Guide.

*CURS-FIELD

Format/length:	I4
Content modifiable:	No

This system variable contains the internal identification of the field in which the cursor is currently positioned.

*CURS-FIELD cannot be used by itself, but only in conjunction with the POS function. You may use them to check if the cursor is currently positioned in a specific field and have processing performed depending on that condition. See the POS function for details.

If the cursor is not in a field or if no REINPUT is possible, *CURS-FIELD contains 0.

Note:
The value of *CURS-FIELD serves only as internal identification of the field and cannot be used for arithmetic operations.
If *CURS-FIELD identifies an occurrence of an X-array (an array for which at least one bound in at least one dimension is specified as extensible), the value of *CURS-FIELD may change after the number of occurrences for a dimension of the array has been changed using the EXPAND, RESIZE or REDUCE statements.

See also Dialog Design, Field-Sensitive Processing, in the Natural Programming Guide.

*CURS-LINE

Format/length: P3

Content modifiable: Yes (however, a negative value or 0 must not be assigned)

This system variable contains the number of the line in which the cursor is currently positioned.

The cursor position is defined within the current active window, regardless of its physical placement on the screen, starting with position 1/1 from the upper left corner of a *logical* page.

Note:
The message line, function-key lines and infoline/statistics line are not counted as data lines on the screen.

*CURS-LINE may also contain one of the following values:

Value	Cursor Position
0	On the top or bottom horizontal frame line of a window.
-1	On the Natural message line.
-2	On the Natural infoline/statistics line.
-3	On the upper function-key (number) line.
-4	On the lower function-key (name) line.

If the value of *CURS-COL is negative, which indicates that the cursor is outside the active window, *CURS-LINE will also contain a negative value. In this case, the *absolute* values of both system variables indicate the position of the cursor on the *physical* screen.

See also Dialog Design, Line-Sensitive Processing, in the Natural Programming Guide.

*CURSOR

Format/length: N6

Content modifiable: No

This system variable contains the position of the cursor on the input screen at the time the ENTER key or a function key is pressed.

Note:
Instead of *CURSOR, it is recommended that the system variables *CURS-LINE and *CURS-COL be used. *CURSOR only continues to be available for compatibility with previous Natural versions.

*LINE-COUNT

Format/length:	P5
Content modifiable:	No

This system variable contains the line number of the current line within the current page.

This variable is used by Natural to determine the line number for the next line of the report.

The value of *LINE-COUNT is incremented by "1" for each line to be output. The value is updated during the execution of a WRITE, SKIP, DISPLAY, PRINT or INPUT statement and contains the number of the last line on the page that has been output.

An EJECT or NEWPAGE statement causes *LINE-COUNT to be reset to "1" (except in the case of NEWPAGE WITH TITLE, where the value of *LINE-COUNT depends on the number of lines output as title).

The maximum line number permitted is 250.

If multiple reports are being produced by the program, (rep) in parentheses after *LINE-COUNT is used to specify the report identification for which the current line number is being requested.

*LINESIZE

Format/length:	N7
Content modifiable:	No

This system variable contains the physical line size of the I/O device from which Natural was invoked (if the TP system is able to provide such).

*LOG-LS

Format/length:	N3
Content modifiable:	No

This system variable contains the line size of the logical page that is output with the primary report.

*LOG-LS is only applicable to the primary report, not to any additional report.

*LOG-PS

Format/length:	N3
Content modifiable:	No

This system variable contains the page size of the logical page that is output with the primary report.

*LOG-PS is only applicable to the primary report, not to any additional report.

*PAGE-NUMBER

Format/length: P5

Content modifiable: Yes

This system variable contains the current value for page number of an output report.

If multiple reports are being produced by the program, *(rep)* in parentheses after *PAGE-NUMBER is used to specify the report identification for which the current page number is being requested.

This variable is defined by Natural at the time formatting for the report is started. Therefore, the parameter has no meaning until the first FORMAT, WRITE, or DISPLAY statement for any given report has been issued. This variable may be modified by a Natural program.

This variable is used by Natural to determine the page number for the next page of the report. The value is always incremented by 1 for the next page initiated by WRITE, DISPLAY, SKIP or NEWPAGE statements. EJECT does not cause *PAGE-NUMBER to be incremented.

*PAGESIZE

Format/length: N7

Content modifiable: No

This system variable contains the physical page size of the I/O device from which Natural was invoked (if the TP subsystem is able to provide such).

*PF-KEY

Format/length: A4

Content modifiable: No

This system variable contains the identification of the key which was pressed last.

*PF-KEY can contain one of the following values:

PA1 to **PA3**	Program Attention keys 1 to 3
PF1 to **PF48**	Program Function keys 1 to 48
ENTR	ENTER key
CLR	CLEAR key
PEN	Light pen
PGDN	PAGE DOWN key (only under UNIX).
PGUP	PAGE UP key (only under UNIX).

*PF-KEY only contains the identification of a key if that key has been made sensitive on that level; otherwise *PF-KEY will contain "ENTR".

Note:
When you compare the content of *PF-KEY with a range of values, remember that *PF-KEY contains an alphanumeric value.

See also Dialog Design, Processing Based on Function-Keys, in the Natural Programming Guide.

*PF-NAME

Format/length:	A10
Content modifiable:	No

This system variable contains the name of the function key that was pressed last, that is, the name as assigned to the key with the NAMED clause of the SET KEY statement.

This allows you to perform processing depending on a specific function name, not a specific key. For example, if you wish to allow users to invoke help by pressing either PF1 or PF13, you assign the name "HELP" to the keys PF1 and PF13 and make the invoking of help dependent on *PF-NAME=’HELP’: the help will then be invoked no matter whether the user presses PF1 or PF13 to invoke it.

See also Dialog Design, Processing Based on Function-Key Names, in the Natural Programming Guide.

*WINDOW-LS

Format/length:	N3
Content modifiable:	No

This system variable contains the line size of the logical window (without frame). See also the DEFINE WINDOW statement.

*WINDOW-POS

Format/length:	N6
Content modifiable:	No

This system variable contains the position which corresponds to the upper left corner of the window. See also the DEFINE WINDOW statement.

The position is counted in characters across multiple lines, beginning with "0" (upper left corner).

*WINDOW-PS

Format/length:	N3
Content modifiable:	No

This system variable contains the page size of the logical window (without frame). See also the DEFINE WINDOW statement.

Natural Environment Related System Variables

The following Natural system variables are covered:

- *DEVICE
- *GROUP
- *HARDCOPY
- *INIT-USER
- *LANGUAGE
- *NATVERS
- *NET-USER
- *PARM-USER
- *PATCH-LEVEL
- *PID
- *SCREEN-IO
- *SERVER-TYPE
- *UI
- *USER
- *USER-NAME

Content modifiable:
In the following text, this indicates whether in a Natural program you can assign another value to the system variable, that is, overwrite its content as generated by Natural.

---

*DEVICE

Format/length: A8

Content modifiable: No

This system variable contains the device type/mode from which Natural has been invoked. It may contain one of the following values:

Value	Description
BATCH	Batch mode.
COLOR	3279 compatibility. 3278 screen device (device with extended attribute support).
VIDEO	3270 screen device, PC screen device, DEC VT terminal or any type of UNIX terminal.
TTY	Teletype or other start/stop device.
PC	Usage of Natural Connection has been activated (by profile parameter PC=ON or terminal command %+).
BTX	BTX device.
SPOOL	3270 printer device.
ASYNCH	Asynchronous session.

*GROUP

Format/length: A8

Content modifiable: No

This system variable is applicable under Natural Security only. It contains the ID via which a user is logged on to a protected library, that is, the ID via which the user is linked to the library. This may be either the ID of the group via which the user is linked or the user's own ID (if he or she is linked directly).

*GROUP will be blank

- in the case of a logon to an unprotected library (where no link is used),
- if Natural Security is not active.

*HARDCOPY

Format/length: A8

Content modifiable: Yes

This system variable contains the name of the hardcopy device which will be used when the terminal command %H is used.

*INIT-USER

Format/length: A8

Content modifiable: No

This system variable contains the user ID of the user.

*INIT-USER under UNIX and Windows

*INIT-USER contains the value of the profile parameter USER in the parameter file used.

If no value is specified for the USER parameter, *INIT-USER contains the user ID used to log in to UNIX, or under Windows the ID you were requested to enter when starting Natural (default value: "SAGPC").

*INIT-USER under UTM

*INIT-USER contains the user ID defined for the UTM application; if no user IDs are defined for the UTM application, *INIT-USER is identical to *INIT-ID.

*INIT-USER under TIAM

The value of *INIT-USER is determined by the parameter USERID in macro NATTIAM: If USERID=USER or NO (default), it contains the BS2000/OSD job name specified with the LOGON command; if no BS2000/OSD job name has been specified, *INIT-USER contains the same as with USERID=SYSTEM (or YES), that is, the BS2000/OSD user ID.

*INIT-USER in batch mode on mainframe computers

*INIT-USER contains name of the job under which the Natural session is running.

*INIT-USER in batch mode under OS/390

The value of *INIT-USER is determined by the parameter USERID in the Natural/OS/390 batch interface (macro NTOS): If USERID=YES, the value will be taken from the security access control block (ACEE) of the security package (for example, RACF or ACF2) being used. If no security package is used, the value will be taken from the USER parameter in the job card. If no USER parameter is specified, the value will be same as with USERID=NO, that is, the name of the job under which the Natural session is running.

*LANGUAGE

Format/length:	I1
Content modifiable:	Yes

This system variable contains the language indicator (language code). This language indicator is used for edit masks of date fields, Natural error messages and user error messages as used in the statements INPUT and REINPUT.

A one-character code is assigned to each language code; this one-character code is used to replace all "&" in names of language-specific objects (for example, maps, dialogs, helproutines, subprograms).

You can specify up to 60 different language codes. The codes are listed below.

The system variable *LANGUAGE is set by the Natural profile parameter ULANG which determines the language to be used for date edit masks, system messages, user messages, help texts, help routines and multi-lingual maps.

On mainframes, the compiler always uses only the current value of *LANGUAGE to determine the map name. During runtime, Natural for mainframes tries to read the map with the current *LANGUAGE setting first. If not found, then it tries to find the map with the default language.

Under Windows and UNIX, Natural does not differentiate without between compile time and run time. It always tries to read the map with the current value of *LANGUAGE first and if not found, it then tries to find the map with the default language.

For details on how to use language codes, see also Designing User Interfaces in the Natural Programming Guide.

Language Code Assignments

The following languages are assigned to the individual language codes (the right-hand column shows the corresponding one-character codes to be used in names of language-specific maps etc.):

- Left-to-Right Single-Byte Languages with Latin Lower Case
- Left-to-Right Single-Byte Languages without Latin Lower Case
- Bi-directional Single-Byte Languages without Latin Lower Case
- User-Assigned Languages
- Multiple-Byte Languages
- Double-Byte Languages

Left-to-Right Single-Byte Languages with Latin Lower Case

Code	Language	Code in Map Names
1	English	1
2	German	2
3	French	3
4	Spanish	4
5	Italian	5
6	Dutch	6
7	Turkish	7
8	Danish	8
9	Norwegian	9
10	Albanian	A
11	Portuguese	B
12	Chinese Latin (Taiwan)	C
13	Czech	D
14	Slovak	E
15	Finnish	F
16	Hungarian	G
17	Icelandic	H
18	Korean	I
19	Polish	J
20	Romanian	K
21	Swedish	L
22	Croatian	M
23	Catalan	N
24	Basque	O
25	Afrikaans	P

Left-to-Right Single-Byte Languages without Latin Lower Case

Code	Language	Code in Map Names	
26	Bulgarian	Q	
27	Greek	R	
28	Japanese (Katakana)	S	
29	Russian	T	
30	Serbian	U	

Bi-directional Single-Byte Languages without Latin Lower Case

Code	Language	Code in Map Names
31	Arabic	V
32	Farsi (Iran)	W
33	Hebrew	X
34	Urdu (Pakistan)	Y
35	(reserved for future use)	Z
36	(reserved for future use)	a
37	(reserved for future use)	b
38	(reserved for future use)	c
39	(reserved for future use)	d
40	(reserved for future use)	e

User-Assigned Languages

Code	Language	Code in Map Names
41	(free for you to assign a language)	f
42	(free for you to assign a language)	g
43	(free for you to assign a language)	h
44	(free for you to assign a language)	i
45	(free for you to assign a language)	j
46	(free for you to assign a language)	k
47	(free for you to assign a language)	l
48	(free for you to assign a language)	m
49	(free for you to assign a language)	n
50	(free for you to assign a language)	o

Multiple-Byte Languages

Code	Language	Code in Map Names
51	Hindi	p
52	Malayan	q
53	Thai	r
54	(reserved for future use)	s
55	(reserved for future use)	t
56	(reserved for future use)	u

Double-Byte Languages

Code	Language	Code in Map Names
57	Chinese (People's Republic of China)	v
58	Chinese (Republic of China)	w
59	Japanese (Kanji)	x
60	Korean	y

*NATVERS

Format/length: A8

Content modifiable: No

This system variable contains the Natural version excluding the patch level information in the format *rr.vv.ss*, where *r*=release, *v*=version, *s*=system maintenance level (example: 04.01.01). The patch level information is contained in the variable *PATCH-LEVEL.

*NET-USER

Format/length: A253

Content modifiable: No

On UNIX and mainframe computers, the value of *NET-USER is identical to the one of *USER.

On Windows, the value of *NET-USER contains the complete user ID consisting of the domain name and the actual user ID.

The following considerations apply to a NaturalX server (Windows platforms only).

When a NaturalX server receives an authenticated request, the user ID of this request is passed to the server and placed into *NET-USER. (The DCOM function "CoQueryClientBlanket" is used for this purpose.)

After the NaturalX server has processed the request, *NET-USER is reset to the value it contained before the request.

A request which is not authenticated has no effect on *NET-USER.

*PARM-USER

Format/length: A253

Content modifiable: No

This system variable contains the name of the parameter module currently in use (if PARM=*name* has not been specified as a dynamic parameter, *PARM-USER contains blanks).

*PATCH-LEVEL

Format/length: A8

Content modifiable: No

This system variable contains the current patch-level number as a string value. See also *NATVERS.

*PID

Format/length: A32

Content modifiable: No

This system variable contains the current process ID as a string value.

On mainframe computers, this system variable contains a unique session ID.

*SCREEN-IO

Format/length: L

Content modifiable: No

This system variable indicates whether a screen I/O is possible or not.

It can contain one of the following values:

TRUE	Screen I/O is possible.
FALSE	Screen I/O is not possible.

In an interactive Natural session, *SCREEN-IO is initialized with "TRUE".

If Natural was started as a DB2 Stored Procedures server (*SERVER-TYPE = DB2-SP) or as RPC server (*SERVER-TYPE = RPC) *SCREEN-IO is set to "FALSE".

If Natural was started on a Windows platform as DCOM server (*SERVER-TYPE = DCOM), *SCREEN-IO is set to "FALSE", while the server is executing a method called by COM/DCOM.

When *SCREEN-IO is set to "FALSE" and a statement which requires user interaction is executed, Natural issues error NAT0723.

*SERVER-TYPE

Format/length: A32

Content modifiable: No

This system variable indicates the server type Natural has been started as.

It can contain one of the following values:

DB2-SP	Natural DB2 Stored Procedures server.
DCOM	NaturalX DCOM server. Applies to Natural for Windows only.
DEVELOP	Natural development server.
RPC	Natural RPC server.

If Natural is not started as a server, *SERVER-TYPE is set to blanks.

Note:
*SERVER-TYPE refers to Natural as a whole, **not** to the Natural program currently being executed (which may run as a client program or as a server program within a server Natural).

*UI

Format/length: A16

Content modifiable: No

This system variable indicates the type of user interface being used:

CHARACTER	Character-oriented user interface.
GUI	Graphical user interface.

*USER

Format/length: A8

Content modifiable: No

This system variable contains the user ID as taken from the Natural Security logon.

If the profile parameter AUTO=ON (Automatic Logon) is set or if Natural Security is not active, the value of *USER is identical to that of *INIT-USER.

*USER-NAME

Format/length: A32

Content modifiable: No

If Natural Security is installed, this variable contains the name of the user who is currently logged on to Natural.

If Natural Security is not active, the default is "SYSTEM".

System Environment Related System Variables

The following Natural system variables are covered:

- *HARDWARE
- *HOSTNAME
- *INIT-ID
- *INIT-PROGRAM
- *MACHINE-CLASS
- *OPSYS
- *OS
- *OSVERS
- *TP
- *TPSYS
- *TPVERS
- *WINMGR
- *WINMGRVERS

Content modifiable:
In the following text, this indicates whether in a Natural program you can assign another value to the system variable, that is, overwrite its content as generated by Natural.

---

*HARDWARE

Format/length:	A16
Content modifiable:	No

This system variable contains the name of the hardware platform on which Natural is running (for example, AIX). This value is supplied by the operating system.

*HOSTNAME

Format/length:	A64
Content modifiable:	No

The name of the machine Natural runs on.

*INIT-ID

Format/length:	A8
Content modifiable:	No

*INIT-ID under UNIX

*INIT-ID contains the ID of the device from which Natural was invoked.

*INIT-ID under Windows 98 and Windows NT/2000

*INIT-ID contains the value "PC_WIN".

*INIT-ID on mainframe computers

*INIT-ID contains the terminal ID (defined according to the conventions of the TP system) of the terminal from which Natural was invoked.

*INIT-ID on mainframe computers in batch mode

*INIT-ID contains the step name of the Natural job.

*INIT-ID for an asynchronous Natural session under Com-plete or UTM

*INIT-ID contains the terminal ID of the task that has started the asynchronous session.

*INIT-ID for an asynchronous session under CICS

*INIT-ID contains the CICS task number of the asynchronous task.

*INIT-PROGRAM

Format/length:	A8
Content modifiable:	No

This system variable contains the name of the program (transaction) that is currently executing as Natural.

*INIT-PROGRAM in batch mode under OS/390

*INIT-PROGRAM contains the name of the job under which the Natural session is running.

*INIT-PROGRAM under UNIX and Windows

*INIT-PROGRAM contains the value "Natural".

*MACHINE-CLASS

Format/length:	A16
Content modifiable:	No

This system variable contains the name of the machine class on which Natural is running.

It can contain one of the following values:

MAINFRAME
PC
UNIX
VMS

*OPSYS

Format/length: A8

Content modifiable: No

This system variable contains the Natural name of the operating system that is being used.

It can contain one of the following values:

ATT_OSX	FACOM/XA	RS_6000
AVIION	FUJI M73	SCO
BS2000	HP_HPUX	SINIX_52
BS2/XS	MSDOS	SINIX_54
BULL/BOS	MS_OS/2	SUN_SOLA
CMS	MVS/ESA	SUN_SUNO
CMS/ESA	MVS/XA	UNISYS 5
DEC-OSF/	NCR 3000	UNISYS 6
DOS/VS	OS	VSE/ESA
DPS300	OS/400	WNT-AXP
DRS 6000	OVMS/AXP	WNT-X86
FACOM	OVMS/VAX	

Note:
Instead of *OPSYS, it is recommended that the system variables *MACHINE-CLASS, *HARDWARE and *OS be used, as they allow a more precise distinction of the environment in which Natural is running.

*OS

Format/length: A32

Content modifiable: No

This system variable contains the name of the operating system under which Natural is running (for example, HP-UX). This value is supplied by the operating system and may be subject to change.

*OSVERS

Format/length:	A16
Content modifiable:	No

This system variable contains the version number of the operating system under which Natural is running. This value is supplied by the operating system and may be subject to change.

*TP

Format/length:	A8
Content modifiable:	No

It contains the name of the TP subsystem under which Natural is running. This value is supplied by the operating system and may be subject to change.

*TPSYS

Format/length:	A8
Content modifiable:	No

This system variable contains the Natural name of the TP monitor or environment that is being used.

It can contain one of the following values:

AIM/DC
CICS
COMPLETE
IMS/DC
OS/400
SERVSTUB (NDV server)
TIAM
TSO
TSS
UTM
VM/CMS

In batch mode, *TPSYS will be blank.

If no TP monitor is used, *TPSYS contains the value "NONE".

*TPVERS

Format/length: A8

Content modifiable: No

It contains the version of the TP subsystem under which Natural is running. This value is supplied by the operating system and may be subject to change.

*WINMGR

Format/length: A16

Content modifiable: No

If a graphical user interface is used, this system variable contains the name of the window manager being used (for example, MOTIF or PM).

If a character-oriented user interface is used, *WINMGR will be blank.

The type of user interface is indicated by the value of the system variable *UI.

*WINMGRVERS

Format/length: A16

Content modifiable: No

If a graphical user interface is used, this system variable contains the version number of the window manager being used.

If a character-oriented user interface is used, *WINMGRVERS will be blank.

The type of user interface is indicated by the value of the system variable *UI.

XML Related System Variables

The following Natural system variables are covered:

- *PARSE-COL
- *PARSE-LEVEL
- *PARSE-NAMESPACE-URI
- *PARSE-ROW
- *PARSE-TYPE

These system variables, which are available when using the PARSE statement, are only valid in the current loop context.

Content modifiable:
In the following text, this indicates whether in a Natural program you can assign another value to the system variable, that is, overwrite its content as generated by Natural.

---

*PARSE-COL

This system variable is only available under Windows and UNIX.

Format/length:	I4
Content modifiable:	No

This Natural system variable is automatically created for each PARSE statement issued.

It specifies the column where the parser is currently working at.

The system variable *COUNTER usable with the database READ loop is also available for the PARSE XML loop.

*PARSE-LEVEL

This system variable is only available under Windows and UNIX.

Format/length:	I4
Content modifiable:	No

This Natural system variable is automatically created for each PARSE statement issued.

It specifies the level of currently nested elements.

*PARSE-NAMESPACE-URI

This system variable is only available under Windows and UNIX.

Format/length:	A (dynamic)
Content modifiable:	No

This Natural system variable is automatically created for each PARSE statement issued.

It specifies the namespace URI of the current element/attribute, if the element/attributes belong to a namespace. If the NAME (*Operand3*) value is empty, then there is also no namespace and *LENGTH(*field*) is set to 0.

Note:
*LENGTH(*PARSE-NAMESPACE-URI) is a valid statement!

*PARSE-ROW

This system variable is only available under Windows and UNIX.

Format/length:	I4
Content modifiable:	No

This Natural system variable is automatically created for each PARSE statement issued.

It specifies the row where the parser is currently working at.

*PARSE-TYPE

This system variable is only available under Windows and UNIX.

Format/length:	A1
Content modifiable:	No

This Natural system variable is automatically created for each PARSE statement issued.

It specifies the type of the delivered data.

Possible values for ASCII-based systems are:

?	Processing instruction (but not first <?XML ... ?>)
!	Comment.
C	CDATA section.
T	Starting tag.
@	Attribute (or § on mainframes).
/	Closing tag.
$	Parsed data.

